NESSUS/NASTRAN INTERFACE

Prepared by

Southwest Research Institute

Final Report
NASA Contract NAS8-39797
SwRI Project 06-7212

Prepared for

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
George C. Marshall Space Flight Center
Marshall Space Flight Center, Alabama 35812

July 1996

Table of Contents

1.0 Introduction and Overview

The NESSUS probabilistic analysis computer program has been developed with a built-in finite element analysis program NESSUS/FEM. However, the NESSUS/FEM program is specialized for engine structures and may not contain sufficient features for other applications. In addition, users often become well acquainted with a particular finite element code and want to use that code for probabilistic structural analysis. For these reasons, this work was undertaken to develop an interface between NESSUS and NASTRAN such that NASTRAN can be used for the finite element analysis and NESSUS can be used for the probabilistic analysis. In addition, NESSUS was restructured such that other finite element codes could be more easily coupled with NESSUS.

NESSUS has been enhanced such that NESSUS will modify the NASTRAN input deck for a given set of random variables, run NASTRAN and read the NASTRAN result. The coordination between the two codes is handled automatically.

The work described here was implemented within NESSUS 6.2 which was delivered to NASA in September 1995. The code runs on Unix machines: Cray, HP, Sun, SGI and IBM.

The new capabilities have been implemented such that a user familiar with NESSUS using NESSUS/FEM and NASTRAN can immediately use NESSUS with NASTRAN. In other words, the interface with NASTRAN has been implemented in an analogous manner to the interface with NESSUS/FEM. Only finite element specific input has been changed.

This manual is written as an addendum to the existing NESSUS 6.2 manuals. We assume users have access to NESSUS manuals and are familiar with the operation of NESSUS including probabilistic finite element analysis. Update pages to the NESSUS PFEM manual are contained in Appendix E.

The finite element features of the code are summarized in Table 1. The probabilistic analysis capabilities are summarized in Table 2.

Table 1. Summary of Finite Element Code Capabilities

- Any quantity in the NASTRAN bulk data deck can be a random variable or affected by a random variable.
- User-written subroutine available for perturbing NASTRAN inputs
- Any NASTRAN solution type can be used.
- Any NASTRAN response which can be written to the output2 file can be returned to NESSUS as the NASTRAN response.
- The identical user-written subroutine, UZFUNC, can be used with NESSUS/FEM or NASTRAN.
- NASTRAN can be run on a machine different than NESSUS (remote host option)
- NASTRAN is called automatically from NESSUS

Table 2. Summary of Probabilistic Code Capabilities
- Advanced Mean Value with iterations, - Monte Carlo and Latin Hypercube sampling.

A schematic of the NESSUS/NASTRAN code is given below.

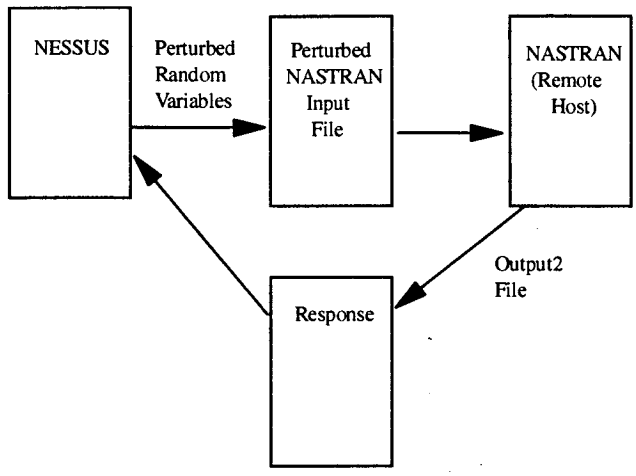


Figure 1. NESSUS/NASTRAN Schematic

2.0 Random Variable Definitions

During a probabilistic analysis, NESSUS requires multiple NASTRAN analyses with each analysis subject to different random variable values. This is accomplished for each analysis by modifying the NASTRAN input deck for a given set of random variables, running NASTRAN and reading the NASTRAN result from the output2 file. Thus, a new NASTRAN input deck must be generated repeatedly according to the random variables. There are two methods to define the random variable input: i) explicitly list the NASTRAN bulk data cards affected by the random variable, or ii) use the user-written subroutine URVDEF. Explicitly listing the NASTRAN bulk data cards is analogous to the approach used for NESSUS/FEM. The URVDEF subroutine is a new option that can be used to define a general relationship between the random variable and the NASTRAN input.

2.1 Explicit listing of the NASTRAN bulk data cards

The definition of the random variables in NESSUS when explicitly listing the NASTRAN bulk data cards is analogous to that when using NESSUS/FEM. The procedure to define a random variable is summarized as:

The `*DEFINE` block in the `*RVDEFINE` section is used as when using NESSUS/FEM.

The random variable name is given.

The mean, standard deviation and distribution type are given.

The NASTRAN card to perturb is given ***exactly*** as in the NASTRAN bulk data deck.

The field number(s) to perturb and the scale factor(s) for the amount of perturbation is given on the next line. (More than one field can be perturbed.)

(Reminder, the total amount of perturbation of a field is the scale factor * the perturbation size defined in the `*PERT` card * the random variable standard deviation)

General format

```
*DEFINE   #
Random Variable Name
mean, standard deviation, distribution type
Unperturbed NASTRAN card
nf1, sf1, nf2, sf2,..., fn, sfn (on same line)
Unperturbed NASTRAN card
nf1, sf1, nf2, sf2,..., fn, sfn (on same line)
```

Continue with each NASTRAN card required for the definition of the random variable.

`nfi` is the field number to change.
`sfi` is the scale factor to change the field.

Input example

For example, the length of a beam is a random variable. The mean length is 10.0 inches and the standard deviation is 1 inch. The beam is defined with 2 beam elements and 3 grid points. The distribution is modeled as Weibull.

For all grid cards, field 4 will be perturbed. Field 4 of the grid 1 card has a scale factor of 0, i.e., it will not be perturbed. Field 4 of the grid 2 card has a scale factor of 0.5 and field 4 of the grid 3 card has a scale factor of 1.0. Grid cards not explicitly defined in the `*DEFINE` block will not be perturbed. Thus, the grid 1 card could be left out from the `*DEFINE` block.

```
*DEFINE   1
LENGTH
10.0      1.0     WEIBULL
grid,  1,  0,  0.0,  0.0,  0.0,  0,  123456
4, 0.0
grid,  2,  0,  5.0,  0.0,  0.0,  0
4, 0.5
grid,  3,  0, 10.0,  0.0,  0.0,  0
4,1.0
```

More than one field of a card can be perturbed. For example, if we wanted to perturb the x and y coordinates of grid 3, we could use

```
grid,  3,  0, 10.0,  0.0,  0.0,  0
4,1.0, 5,0.5
```

How it works

NESSUS modifies the NASTRAN input deck in the following manner. For each NASTRAN card in the `*DEFINE` block,

- compute the perturbed field(s) values according to the expansion point, scale factor, perturbation size and standard deviation,
- locate the unperturbed card in the NASTRAN bulk data deck
- replace the unperturbed card with the perturbed card

Restrictions

NESSUS will match the unperturbed card in the `*DEFINE` block with the cards in the bulk data deck using a character match. The two cards must match **EXACTLY**. (The safest method to develop the `*DEFINE` block is to copy the bulk data card and paste it in the `*DEFINE` block, thus ensuring no differences.)

Any format for the card can be used - free or fixed format. In addition, if using fixed format, large or small format may be used. The only requirement is that the format be the same in the `*DEFINE` block as in the bulk data deck. In fact, different formats can be used in the same input file as long as the formats for any particular card match between the `*DEFINE` block and the bulk data deck. Note, however, that more precision can often be obtained with the free format or large fixed format. For example, when perturbing Young's modulus, the significant figures that can be obtained from an 8 character field (fixed small format) is limited. More significant figures can be obtained from a free or large fixed format.

Only real numbers of the NASTRAN bulk data section can be perturbed.

2.2 User subroutine URVDEF

A user defined subroutine can be used to define the random variable perturbations as an alternative to explicitly listing the NASTRAN bulk data cards to perturb in the `*DEFINE` block. This subroutine approach is newly developed and not available when using NESSUS/FEM. This method can be used to define an arbitrary relationship between the random variables and the NASTRAN input; whereas explicitly listing the NASTRAN bulk data cards can only define a linear relationship. A user random variable is selected by identifying the define block type as `USER`.

General Format

```
*DEFINE    #
Random Variable Name
mean, standard deviation, distribution type
USER
```

How it works

All USER defined random variables are handled with a single call to the user subroutine URVDEF. This occurs after all explicitly listed random variables have been processed. The actual current values of all random variables are provided in URVDEF. The user identifies the card and field numbers that need to be replaced and the value of the fields. The perturbation of the USER random variables is not cumulative as for the explicitly listed random variables, meaning any fields defined using the subroutine URVDEF will overwrite any perturbations of these fields by any explicitly listed random variables.

The following steps are required for coding URVDEF for USER defined random variables

1. Identify the card to match and its length
2. Define the number of fields to replace and the field numbers
3. Define the field values to be replaced
4. Call subroutine CHGNCARD to make the changes in the perturbed NASTRAN deck
5. Repeat as needed

Restrictions

The card to match must be unique in the NASTRAN bulk data input. NESSUS will find the first occurrence of the card. Most cards in the NASTRAN input are unique but if they are not, the user must add unique characters in field 10 of the card to match.

It is assumed that the matching characters begin in column 1. Subroutine URVDEF will not match the characters from the middle of the card. The characters to match must start from column 1.

Continuation cards for the large fixed format cannot presently be perturbed because of the '*' character in column 1. This restriction will be removed at a later date. Continuation cards for small fixed format and free format can be perturbed as long as a '+' character is in column 1 of the continuation card.

Input Example

An example problem is used to further explain the use of URVDEF. The problem is a simply supported beam with a hollow rectangular cross-section. Figure 2 shows the problem set up and Table 3 lists the random variable statistics. The beam cross-sectional width, height and thickness are random variables but are not direct NASTRAN input quantities. These random variables

define the moments of inertia, cross-sectional area, and torsional constant defined on the PBAR card. The subroutine URVDEF relates the input random variables to NASTRAN input quantities for the three USER random variables. The elastic modulus, load, and length are defined by explicitly listing the NASTRAN cards. The required PFEM input and the subroutine are explained below.

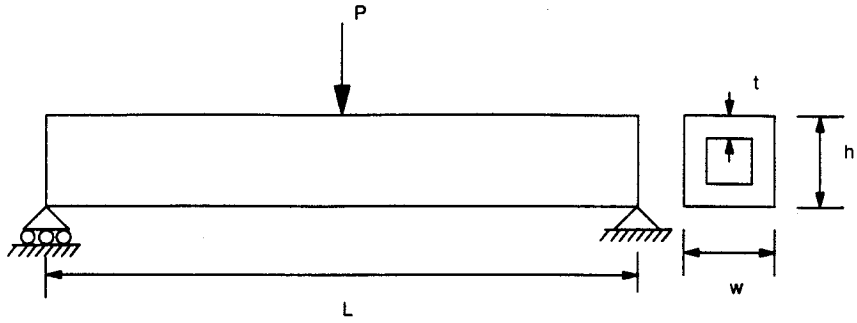


Figure 2. Simply supported beam using user defined random variables

Table 3. Random Variables for Simply Supported Beam Example			
Random Variable	Mean	Standard Deviation	Distribution*
THICK	0.1 in	0.01 in	NORMAL
WIDTH	1.0 in	0.1 in	NORMAL
HEIGHT	1.0 in	0.1 in	NORMAL
EMOD	30.E6 psi	30.E5 psi	NORMAL
LOAD	100.0 in	10.0 in	NORMAL
LENGTH	10.0 in	1.0 in	NORMAL

*Any distribution type can be used.

The random variables are defined in the `*RVDEFINE` section of the PFEM input as listed below.

```
*RVDEFINE
*DEFINE     1
THICK
  0.1   0.01   NORMAL
USER
*DEFINE     2
WIDTH
  1.0   0.1    NORMAL
USER
*DEFINE    3
HEIGHT
  1.0   0.1    NORMAL
USER
*DEFINE     4
EMOD
```

```
  30.0E6   30.0E5 NORMAL
mat1,1,3.000E7,0.3, .1000
3,1.0
*DEFINE      5
LOAD
  100.0   10.0   NORMAL
force, 1, 6, 0,   100.0, 0.0, -1.0, 0.0
5,1.0
*DEFINE      6
LENGTH
    10.0   1.0   NORMAL
grid,   1,   0,   0.0,   0.0,   0.0,   0
4,0.0
grid,   2,   0,   1.0,   0.0,   0.0,   0
4,0.1
grid,   3,   0,   2.0,   0.0,   0.0,   0
4,0.2
grid,   4,   0,   3.0,   0.0,   0.0,   0
4,0.3
grid,   5,   0,   4.0,   0.0,   0.0,   0
4,0.4
grid,   6,   0,   5.0,   0.0,   0.0,   0
4,0.5
grid,   7,   0,   6.0,   0.0,   0.0,   0
4,0.6
grid,   8,   0,   7.0,   0.0,   0.0,   0
4,0.7
grid,   9,   0,   8.0,   0.0,   0.0,   0
4,0.8
grid,  10,   0,   9.0,   0.0,   0.0,   0
4,0.9
grid,  11,   0,  10.0,   0.0,   0.0,   0
4,1.0
```

The URVDEF user subroutine is listed below describing the relationship of the random variables to the NASTRAN input

```
      subroutine urvdef(ICREAD,ICONSL,ILPRNT,ISCRH2,ISCRH3,NLINE,ILINE,
     +                  NRV,XSTAR,IERR)
      implicit double precision (a-h,o-z)
C
C Purpose:
C     Allows user defined random variables that may not
C     be direct finite element quantities.
C
C What it does:
C     Provides a place for the user to define the criteria
c     for matching a line in the input file and replacing
C     specified fields with specified values.  These values
C     can be a random variable or a functional combination
C     of any of the random variables.
```

```
C
C User defined random variables are handled after all other
C DEFINE blocks
C
C Limitations:
C     This routine will replace the specified field with the
C     prescribed value.  The results are not cumulative as
c     for the general define block.
C
C Arguments(S-SENT, R-RETURNED, L-LOCAL):
C    ICREAD       - S - PERTURBED NASTRAN INPUT DECK UNIT
C    ICONSL       - S - SCREEN UNIT
C    ILPRNT       - S - OUTPUT FILE UNIT
C    ISCRH2       - S - PERTURBED NASTRAN INPUT DECK, unit 80
C    ISCRH3       - S - ORIGINAL NASTRAN INPUT DECK, unit 81
C    NLINE        - S - NUMBER OF LINES IN Nastran input deck
C    XSTAR        - S - RV VALUES
C    IERR         - R - ERROR FLAG
C    LENGTH_CARD  - L - LENGTH OF CARD TO MATCH
C    MATCH_CARD   - L - STRING TO MATCH
C    NUM_FIELDS   - L - NUMBER OF FIELDS TO REPLACE
C    IREPLACE     - L - FIELD POSITIONS TO REPLACE
C    RVALUE       - L - FIELD VALUE TO BE REPLACED
C
C Steps for defining user random variables:
C    1) Identify the card to match in MATCH_CARD
C       and the length of MATCH_CARD in LENGTH_CARD
C    2) Define the number of fields to replace in NUM_FIELDS
C       and the field numbers to replace in IREPLACE(i)
C    3) Define the field values to be replaced in RVALUE(i)
C    4) Call CHGNCARD to make the change in the perturbed deck
C        repeat as needed for each card
C
C

      character*80 match_card
      dimension xstar(nrv)
      dimension ireplace(8), rvalue(8)
C
C This routine was developed for a rectangular hollow
C simply supported beam with a point load at mid span.
C
C The random variables are
C  1 thickness
C  2 width
C  3 height
C  4 length
C  5 elastic modulus
C  6 force
C
C The first three random variables are not direct NASTRAN input
C quantities but must be functionally combined for A, I1, I2, J.
C
```

```
C--define the card to match
c--the match_card must be unique
C
      match_card = 'pbar, 1, 1'
      length_card = 10
C
C--define the fields to replace
C  replace fields 4,5,6,7
C
      num_fields = 4
      ireplace(1) = 4
      ireplace(2) = 5
      ireplace(3) = 6
      ireplace(4) = 7
C
C--define the field values
C
      t  = xstar(1)
      w  = xstar(2)
      h  = xstar(3)
      rl = xstar(4)
      e  = xstar(5)
      p  = xstar(6)
c---Area
      rvalue(1) = w*h - (w-2.0D0*t)*(h-2.0D0*t)
c---I1
      rvalue(2) = (w*h**3 - (w-2.0D0*t)*(h-2.0D0*t)**3)/12.0D0
c---I2
      rvalue(3) = (h*w**3 - (h-2.0D0*t)*(w-2.0D0*t)**3)/12.0D0
c---J
      rvalue(4) = (2.0D0*t*(w*h)**2)/(w+h)

C--replace fields on NASTRAN card with perturbed values
      call chngcard(match_card,length_card,iscrh2,iscrh3,nline,iline,
     &                    num_fields,ireplace,rvalue,ierr)
      return
      end
```

Restrictions

The perturbation of the USER random variables is not cumulative as for the general random variables, meaning any fields defined using the subroutine URVDEF will overwrite any perturbations of these fields by any explicitly listed random variables.

3.0 Defining the NASTRAN Response

The response from the NASTRAN analysis is selected using the `*RESPONSE` card in the PFEM input section of NESSUS. NESSUS expects to read the NASTRAN response from the output2 file. Any quantity can be returned to NESSUS as the NASTRAN response if it is written

to the output2 file. The NESSUS CVARIABLE option is not currently supported when using NASTRAN to generate the response.

3.1 MSC/NASTRAN output2 file

The MSC/NASTRAN output2 file consists of data blocks containing the finite element input and output quantities. Each data block contains specific data such as displacements or strains. These data blocks are defined by a data block name. The format of the data block can be different for each type of quantity. The output2 file can be generated using predefined data blocks or customized by the user. See the MSC/NASTRAN DMAP Module Dictionary for further information about the format of the output2 file.

The output2 file is not generated automatically by NASTRAN. The user must command NASTRAN to generate the output2 file using the following card in the bulk data section of the NASTRAN input:

```
PARAM,POST,-1
```

The output2 file must be assigned a unit number and file name using the following statement in the file management section of the NASTRAN input:

```
ASSIGN OUTPUT2 = 'nasjob.op2', UNIT = 12, STATUS=UNKNOWN
```

The name of the file must be "`nasjob.op2`" in lower case. The unit number should be left at 12.

See the example problem in section 5 for example input of these two cards

3.2 Built in response types

The following NASTRAN response types are currently supported and are chosen using the `*RESPONSETYPE` keyword in the PFEM input.

Quantity	Response Key
Displacement	1
Strain	2
Stress	3
Eigenvalue	31
Natural Frequency (radians)	35
Natural Frequency (cycles)	36

Displacements

A displacement response is selected using response type key 1. The displacement component is selected using the `*COMPONENT` keyword in the MVDEFINE and AMVDEFINE sections. The displacement components are 1,2,3 for the translational components and 4,5,6 for rotations if supported by the element type.

Stress/Strain

A strain or stress response is selected using response type key 2 and 3, respectively. Stress and strain components are selected using the *COMPONENT keyword in the MVDEFINE and AMVDEFINE sections of the PFEM input. Stress and strain components are selected based on the NASTRAN element stress/strain item codes (MSC/NASTRAN Quick Reference Guide, Appendix A). Listed below are the stress/strain item codes for the QUAD4 element. As an example, to chose the von Mises stress at Z1 for a QUAD4 element, the component number would be 9.

Element Name (Code)	Real Stresses or Strains	
	Item Code	Item
QUAD4 (33) Linear	2	Z1 = Fiber Distance 1
	3	Normal x at Z1
	4	Normal y at Z1
	5	Shear xy at Z1
	6	⊗ Shear Angle ar Z1
	7	Major principal at Z1
	8	Minor principal at Z1
	9	von Mises or Maximum Shear at Z1
	10	Z2 = Fiber Distance 2
	11	Normal x at Z2
	12	Normal y at Z2
	13	Shear xy at Z2
	14	Shear Angle at Z2
	15	Major principal at Z2
	16	Minor principal at Z2
	17	von Mises or Maximum Shear at Z2

Page A-11, Appendix A of the "MSC/NASTRAN Quick Reference Guide," Version 68, Michael Reymond and Mark Miller, Editors, The MacNeal-Schwendler Corporation, February 1994.

Mode/Subcase selection

The PFEM *CONDITION keyword is used to select the subcase number or mode number.

Node/Element selection

The node or element selection for the NASTRAN response is chosen using the MVDEFINE/AMVDEFINE *NODE or *ELEMENT keyword. The input format has been

improved in terms of user friendliness. The updated NESSUS manual pages are provided in Appendix E.

3.3 Defining other responses

Other NASTRAN responses can be added to the interface capabilities. Currently, the default output2 file generated using the `PARA, POST, -1` statement contains many results of interest. To add additional response types to the interface, the data block name in the output2 file must be identified that contains the response quantity of interest. There is not a published manual defining these data block names, but MSC can assist with identifying the required data block names and format.

Several routines are used to read the output2 file.

MSCPOST - controls reading the output2 file
IHEADR - reads the data block header data (provided with NASTRAN data).
IOPEN - reads the data block label (provided with NASTRAN distribution)
IREAD - reads the data block (provided with NASTRAN distribution).

The MSCPOST routine requires modification when adding a new response. The routine is commented explaining how to add a new response. The general steps for adding a new response are listed here

1. Identify the data block name containing the response and assign a response type key
2. Add an IF-THEN section to match the data block name
3. Identify how the component and subcase/increment relate to the data block contents returned from IREAD
4. Assign the result to XMCRSP and return

If a response quantity is not available in the default output2 file, DMAP ALTERs can be used to write the required data block to the output2 file. (See the MSC/NASTRAN user documentation for further information on DMAP ALTERs)

4.0 Defining the NASTRAN Model in NESSUS

This section defines the NASTRAN analysis model. It identifies where to put the NASTRAN analysis model in the NESSUS input file and how to define the NASTRAN exectution command. The remote host option for executing NASTRAN is described in Section 5.0.

4.1 NASTRAN analysis model

The structure of the NESSUS/NASTRAN PFEM input file is identical to that when using NESSUS/FEM with the exception that no *END is used after the NASTRAN analysis model input and that the input can be imported using an *INCLUDE keyword. The input file has the following structure:

*PFEM
    Probabilistic analysis input
*END
*NASTRAN
    NASTRAN analysis model input (or *INCLUDE filename)
*FPI
    FPI analysis input
*END

When inserting the NASTRAN analysis model directly in the NESSUS input file, the end of the NASTRAN deck is located by finding the `*FPI` keyword. Therefore all lines between `*NASTRAN` and `*FPI` will be used as the template NASTRAN input file. Comments in the NESSUS input file should be avoided between the `*NASTRAN` and `*FPI` keywords.

To include a NASTRAN analysis model, the command has the following form

```
*NASTRAN nastran execution commands
*INCLUDE filename
```

The `*INCLUDE` card must be the next line after the `*NASTRAN` card. The file name is case sensitive.

4.2 Required NASTRAN cards

Several NASTRAN cards are required in the NASTRAN input section to generate the output2 file for response extraction. The cards are defined in Section 3 in more detail with a further explanation about extending the code for other response types. The following card is input in the bulk data section:

```
PARAM,POST, -1
```

This card instructs NASTRAN to write the output2 file.

The following card is required in the NASTRAN file management section:

```
ASSIGN OUTPUT2 = 'nasjob.op2', UNIT = 12, STATUS = UNKNOWN
```

This card assigns a unit number and file name to the output2 file. The name of this file must be `nasjob.op2` and the unit number should be 12.

4.3 NASTRAN execution command

The *NASTRAN card also identifies the execution command for NASTRAN and may be system dependent. The command can be the MSC provided nastran script or a user generated script name for custom execution of NASTRAN. The requirement for the execution command is that control is not returned to NESSUS until NASTRAN has completed the analysis. This can be accomplished by using the MSC provided nastran script with the command line option bat=no or a custom script that waits for the analysis to complete before exiting. The general form of the NASTRAN execution command using the MSC provided nastran script for Unix systems is:

```
*NASTRAN nastran nasjob scr=yes old=no bat=no news=no
```

The second item after the nastran execution command is the NASTRAN input file name. Currently this is required to be nasjob. NESSUS will create the NASTRAN analysis input file using the name nasjob.dat. The interface will not execute if any other name is used.

The command line options are defined as:

scr=yes - delete databases after job ends
old=no - do not save previous output files with version numbers
bat=no - do not execute as a background process
news=no - do not print news and release notes to the .f06 file

The only required command line option is bat=no that prevents NASTRAN from executing as a background process. The other options may be removed but have be found to reduce disk space requirements.

5.0 Using the Remote Host Option for NASTRAN Analysis

NASTRAN has the feature of using a remote host on which to run the NASTRAN job. This feature has been incorporated into the NESSUS/NASTRAN interface. NESSUS can be run on one machine and NASTRAN will be run on a different machine. This is useful if a remote machine has significantly more computational speed than a local machine or if NASTRAN only runs on the remote machine. There are currently two options for remote host execution. The MSC provided nastran script remote host option for use when both local and remote hosts are Unix work stations and a generic option that can be modified for any system. This section includes some general information regarding remote execution of NASTRAN and returning results files to the local machine. Subsections describe further details of the two options for the remote host capability.

The remote host capability allows execution of NASTRAN on a remote computer while NESSUS runs locally. There are three steps required for running NASTRAN on a remote host in the context of probabilistic analysis.

1. Transfer the NASTRAN input file to the remote host

2. Execute NASTRAN on the remote host
3. Transfer the NASTRAN results files from the remote host to the local host

Background Processing:

NESSUS requires that the NASTRAN analysis be complete when control is returned to NESSUS. Both options are required to return control to NESSUS only after the NASTRAN analysis is complete. A modification to the `nastran` script is required to instruct the script to wait until the analysis is complete. The generic script option may require system dependent script code development to determine when the NASTRAN analysis is complete.

Non-binary equivalent remote hosts:

The NESSUS program reads the unformatted output2 file to obtain the response. When the local and remote machines are non-binary equivalent machines then the unformatted output2 cannot be transferred to the local machine and read by NESSUS. The user must perform several steps to circumvent this problem.

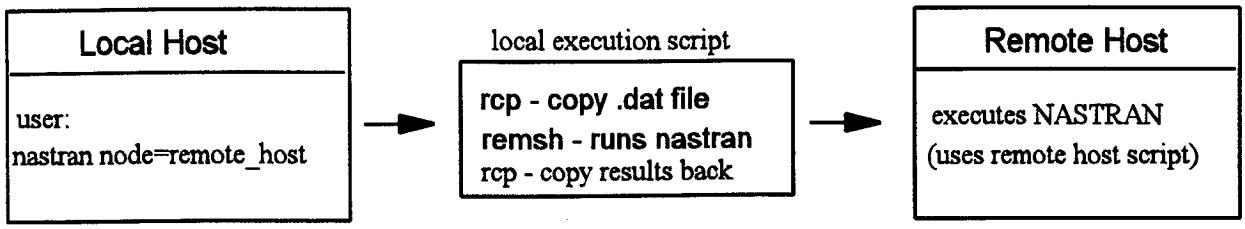
a) First, NASTRAN must be told to write the output2 file in a neutral format (ASCII file). To create a neutral formatted output2 file the statement `FORM=FORMATTED` must be included in the NASTRAN ASSIGN.

b) Second, a Fortran program RCOUT2 must be executed to convert the ASCII file to a unformatted binary file on the local machine. MSC provides the RCOUT2 utility (contained in the MSC util directory).

5.1 NASTRAN script option

On most Unix workstations, NASTRAN is executed by invoking the Bourne shell script (supplied by MSC). Execution options allow remote host execution and the three steps above are handled automatically by the `nastran` script. However, some modifications were needed to copy the output2 file back to the local machine and to force Unix to wait for the completion of the NASTRAN job before returning control to the local machine.

The `nastran` script works as follows. First the `nastran` script interprets the command line options. Upon encountering the option for remote execution, the script creates a local execution script that contains the remote copy (rcp) and remote execution (remsh) commands for the specific job. A diagram of this procedure is shown below. The part of the `nastran` script which creates this local execution script can be found by searching on the text "Build the local execution script" in the `nastran` script.



Several issues arise when running NASTRAN remotely from NESSUS when using the `nastran` script. First, the output2 file is not a standard NASTRAN file and is not automatically returned to the local machine when the execution is completed. Secondly, the script executes NASTRAN as a background process returning control to the caller after the job has been submitted. When running NESSUS, control should be returned to NESSUS when the analysis is complete, not when the job has been submitted as a background process. Finally, the standard form of the output2 file is a binary file and most likely cannot be transferred between different types of computers. These issues can be overcome as described in Section 4.0 and below.

Returning the output2 file:

The output2 file is not a standard NASTRAN results file so it is not returned with the other results files. Also, when executing remotely on an HP workstation, this file is created in the user's home directory not in the /tmp area where the other NASTRAN execution files are located. These problems can be overcome using the following procedures. The output2 file is defined in the NASTRAN input file using the ASSIGN card. By adding the /tmp path to the file name, the output2 file can be found and returned from the remote host. Secondly, using the naming convention specified for the output2 file when using NESSUS (nasjob.op2), the op2 extension can be included in the extension list in the `nastran` script and the file will be returned with the other NASTRAN results files. The ASSIGN statement in the NASTRAN input file will have the following form

```
ASSIGN OUTPUT2 = '/tmp/nasjob.op2', UNIT = 12, STATUS=UNKNOWN
```

If the local and remote machines are non binary equivalent, then the `FORM=FORMATTED` needs to be included as follows

```
ASSIGN OUTPUT2 = '/tmp/nasjob.op2', UNIT = 12, STATUS=UNKNOWN, FORM=FORMATTED
```

The formatted file must then be converted to the unformatted version using RCOUT2 as discussed in Section 4.0.

Background Processing:

NESSUS requires that the NASTRAN analysis be complete when control is returned to NESSUS. The default option for remote host processing using the `nastran` script is to return control from the `nastran` script after the job has been submitted on the remote host. In this case, NESSUS looks for the output2 file before the analysis has completed. To rectify this, a

modification has been made to the nastran script that instructs the script to wait until the process has completed before returning control from the script. Details of this change are contained in the next section.

Changes to the nastran script:

Several changes have been made to the nastran execution script. These changes will need to be made in a local host version of the nastran script. The following changes were made for the HP Unix workstation version of the nastran script. It is assumed that these changes will work on most Unix systems. The Cray system may need additional modifications.

1. The standard Unix wait command is added after submitting the NASTRAN job on the remote host. The portion of the nastran script with the added wait command follows

```
#
# Execute the local execution script.
#
if test -n "${aft}" ; then
     test -n "${MSC_NOEXE}" && echo "MSC_NOEXE set - job $job" >&2 && exit
     echo ${job} | at ${aft}
elif test "${bat}" = "no" ; then
     test -n "${MSC_NOEXE}" && echo "MSC_NOEXE set - job $job" >&2 && exit
     ${job}
else
     test -n "${MSC_NOEXE}" && echo "MSC_NOEXE set - job $job" >&2 && exit
     nohup nice ${job} > /dev/null 2>&1 &
     echo "Background process id = $!"
        wait $!
fi
exit
```

2. The op2 extension is added to the extension list (${exts}) as follows

```
exts="f04 f06 log pch plt out xdb op2 #Files to be versioned
```

Specifying the remote host option

The NASTRAN execution command (after the *NASTRAN card) should include the following statement

```
node=remote_host_address
```

for remote host execution. For example

```
nastran nasjob old=no scr=yes node=talon.struct.swri.edu
```

will execute NASTRAN on the remote host talon.struct.swri.edu.

5.2 Generic remote host option

The generic remote host option uses C-shell scripts that can be customized for any combination of systems. When using this capability, NESSUS will execute a script that controls the execution of NASTRAN instead of executing the MSC supplied `nastran` script. Two scripts are required, the local script and the remote script. The fundamentals of the local script are:

1. Copy NASTRAN input file to remote host
2. Copy remote execution script to remote host
3. Perform a remote execution to start the remote script on the remote host
4. Wait for return of the output2 file
5. Convert the output2 file neutral format to binary format if necessary
6. Return control to NESSUS

The generic required steps for the remote script are as follows

1. Submit the NASTRAN job
2. Copy the output2 file/diagnostic messages back to the local machine

The scripts would need to be customized for each system identifying the local and remote hosts, commands for remote copying specific to the system, and the NASTRAN execution procedures. For example, the execution of NASTRAN on a Cray requires the use of the QSUB command which submits a shell script for executing NASTRAN.

6.0 Example Problem

An example problem is given below to demonstrate the input format. The example problem is to determine the cumulative distribution function of the tip displacement of a cantilever beam subject to midspan and end loads. There are three random variables, the load, Young's modulus, and the beam length.

Force random variable

The two forces on the beam are part of a ramp loading and are defined using NASTRAN force cards. As the ramp loading changes, the forces change with the middle force 1/2 of the end force. The scale factor on the midspan node is 1/2 of the end load, thus simulating a ramp loading. That is, the ratio of the midspan to end load will remain 1/2. The small fixed format is used for this random variable. The `*DEFINE` block is

```
*DEFINE     1
LOAD
100.0       10.0      NORMAL
force             1         3         0   100.0        0.0      -1.0       0.0
5, 1.0
force             1         2         0    50.0        0.0      -1.0       0.0
5, 0.5
```

Material property

The elastic modulus of the beam is a random variable and is defined on the MAT1 card. In this case, the random variable is a finite element quantity. The large fixed format is used for this random variable. The large fixed format is necessary to get the required number of significant digits for this variable. The small fixed format, 8 characters, is sufficient for the mean values but not for perturbed values of Young's modulus such as 2.97445E7. This accuracy is required to calculate gradients of the response correctly. The *DEFINE block is

```
*DEFINE     2
MODULUS
30.E6     30.E5         LOGNORMAL
mat1*                     1           3.0+7                                  0.3 tlarge
3, 1.0
```

Length variable

The length of the beam is defined using 3 grid points and 2 elements. As the length random variable changes, the NESSUS modifies the grid points accordingly. The perturbation of the grids is defined such that element 1 and element 2 remain equal in length. Thus, grid 1 is not perturbed, scale = 0.0, grid 2 has a scale factor of 0.5 and grid 3 has a scale factor of 1.0. Free format is used for this random variable. The *DEFINE block is

```
*DEFINE    3
LENGTH
10.0       1.0      WEIBULL
grid,   1,   0,   0.0,   0.0,   0.0,   0,   123456
4, 0.0
grid,   2,   0,   5.0,   0.0,   0.0,   0
4, 0.5
grid,   3,   0,  10.0,   0.0,   0.0,   0
4,1.0
```

Response selection

The response is chosen using the displacement response key defined in Section 3.2. The node and component are selected as for a standard NESSUS/FEM quantity. The keywords are contained in the MVDEFINE and AMVDEFINE sections of the PFEM input and are listed below for this example.

```
*RESPTYPE    1
*NODE
 3 to 3
*COMP 2 2
*COND 1 1
```

NASTRAN input specification

The NASTRAN input can be specified directly within the NESSUS input file, as shown in the example input deck below, or contained in a separate file. If the NASTRAN input is in a separate file, the user specifies the file name using the `*INCLUDE filename` on the next line after the `*NASTRAN` card. An example is given below.

```
*NASTRAN nastran nasjob scr=yes old=no bat=no news=no
*INCLUDE nasinput
```

Example input deck

The complete NESSUS input deck for the example problem is given below. Note, that the NASTRAN input deck is contained within the NESSUS input file. Alternately, the NASTRAN input can exist in an independent file and be referenced using a `*INCLUDE` statement.

```
*PFEM
C  NESSUS/NASTRAN Interface test problem of cantilever beam with midspan
C  and end loads.  Random variables are the loads, elastic modulus and
C  length
C
C      USES FIXED 8 CHARACTER FORMAT
C
C  Note: when defining the random variables, the format is
C  Unperturbed NASTRAN card
C  field no. to perturb, scale factor
C
C PFEM and FPI inputs are the same as in NESSUS 6.1 except for the
C computational random variables
*ZFDEFINE
*COMPUTATIONAL   4 3    <- the 4 signals that NASTRAN is used
1,2,3
*INPUT          test1.fem
*END
*RVDEFINE
C   3 random variables: beam length,  Young's modulus,
C                       and loads
*DEFINE    1
LOAD
100.0     10.0    NORMAL
force          1       3       0  100.0      0.0    -1.0     0.0
5, 1.0
force          1       2       0   50.0      0.0    -1.0     0.0
5, 0.5
*DEFINE    2
MODULUS
30.E6    30.E5       LOGNORMAL
mat1*                  1           3.0+7                                  0.3 tlarge
3, 1.0
*DEFINE   3
LENGTH
10.0     1.0    WEIBULL
grid,  1,  0,  0.0,  0.0,  0.0,  0,  123456
4, 0.0
grid,  2,  0,  5.0,  0.0,  0.0,  0
4, 0.5
grid,  3,  0, 10.0,  0.0,  0.0,  0
4,1.0
*PERT    1
    1   0.1
*PERT    2
    2   0.1
*PERT    3
    3   0.1
*END
*MVDEFINE
```

```
 *DATA 0
 *COND 1 1
 *NODE
 3 to 3
 *PERT 3
 1,2,3
 *PRINT SHORT
 *RANVAR 3
 1,2,3
 *RESPTYPE   1
 *COMP 2 2
*END
 *AMVDEFINE
 *COND 1 1
 *NODE
 3 to 3
 *COMP 2 2
*ITER
 5 0.01
*END
*END
*NASTRAN nastran nasjob scr=yes old=no bat=no news=no
$   Cantilever beam problem
$
ASSIGN OUTPUT2 = 'nasjob.op2', UNIT = 12, STATUS=UNKNOWN
time 130
sol 101
cend
echo  = both
title = Cantilever beam problem
subcase 1
   analysis = statics
   displacement = all
   elforce(print) = all
   load = 1
begin bulk
PARAM    POST    -1
$C----------------------------------
$   BEGIN ACTUAL NASTRAN BULK DATA
$
$
$ Beam elements
$
cbar           1       1       1       2     0.0     1.0     0.0
cbar           2       1       2       3     0.0     1.0     0.0
$
$ Beam element properties
$
pbar, 1, 1, 1.0, 0.0833333, 0.0833333,0.0,.1406
$
$ Grid points
$
grid,  1,  0,  0.0,  0.0,  0.0,  0,  123456
grid,  2,  0,  5.0,  0.0,  0.0,  0
grid,  3,  0, 10.0,  0.0,  0.0,  0
$
$ Material properties (large fixed format)
$23456789112345678921234567893123456789412345678951234567896123456789712345678981
$      |               |               |               |               |       |
mat1*                  1           3.0+7                           0.3 tlarge
*tlarge         0.100000
$
$ Applied loads (small fixed format)
$
$234567|1234567|1234567|1234567|1234567|1234567|1234567|1234567|1234567|
force          1       3       0  100.0      0.0    -1.0     0.0
force          1       2       0   50.0      0.0    -1.0     0.0
$
enddata
*FPI
 MSC/NASTRAN EXAMPLE PROBLEM
*RVNUM    3
*DATASETS   4
*GFUNCTION   1
```

```
*METHOD    0
*ANALTYPE 2
*PRINT    0
*END
*PLEVELS  11
1.E-6  1.E-5  1.E-4 1.E-3 1.E-2  .1 0.501 .9 .99 .999 .9999
*END
```

7.0 Command Summary for Input Deck Preparation

a) Select computational method 4 using the *COMPUTATIONALMETHOD keyword.

b) Define random variables using the *DEFINE card in the *RVDEFINE section of the *PFEM input. Use method 1 or 2, (Section 2).

c) Define response type, i) either built in option or ii) define DMAP alter to write response to output2 file. If using method ii) modify mscpost.f subroutine to correctly read the response, (Section 3).

d) Use the *NASTRAN card to designate the beginning of the NASTRAN input and define the NASTRAN execution command. Note, the input deck NASTRAN runs is the second item on the *NASTRAN card. In the example below, NASTRAN will run the input deck nasjob. This file name is required to be nasjob. NESSUS will copy the perturbed NASTRAN deck to nasjob.dat and NASTRAN will run this input deck. (See Section 4.)

```
*NASTRAN nastran nasjob scr=yes old=no bat=no news=no
$    Cantilever beam problem
$
ASSIGN OUTPUT2 = 'nasjob.op2', UNIT = 12, STATUS=UNKNOWN
```

The ASSIGN card is used to command NASTRAN to generate the output2 file.

d) Include the NASTRAN input within the NESSUS input file or specify the name of the NASTRAN input file using the *INCLUDE keyword.

e) Select the remote host option on the *NASTRAN card if applicable.

APPENDIX A — Code Restrictions

- The `*INCLUDE` fielname option **must** be located immediately after the `*NASTRAN` card if this option is to be used. Reminder, the filename is case sensitive.

- NESSUS will match the unperturbed card in the `*DEFINE` block with the cards in the bulk data deck using a character match. The two cards must match **EXACTLY**.

- Only real numbers of the NASTRAN bulk data section can be perturbed. The fields to be perturbed cannot contain an integer or any of the special NASTRAN characters %, *,), or =.

- Continuation cards for the large fixed format cannot presently be perturbed because of the '*' character in column 1. This restriction will be removed at a later date. Continuation cards for small fixed format and free format can be perturbed as long as a '+' character is in column 1 of the continuation card.

- The assign card must use nasjob.op2 in lower case as the output2 file name.

- The NASTRAN response must be written to the output2 file and NESSUS must be programmed to read the response. The user must modify NESSUS if a response other than the preprogrammed response is desired.

- The NESSUS CVARIABLE option is not supported when using NASTRAN for the finite element analysis.

- The perturbation of the USER random variables is not cumulative as for the general random variables, meaning any fields defined using the subroutine URVDEF will overwrite any perturbations of these fields by any explicitly listed random variables.

- The card to match in subroutine URVDEF must be unique in the NASTRAN bulk data input. NESSUS will find the first occurrence of the card. Most cards in the NASTRAN input are unique but if they are not, the user must add unique characters in field 10 of the card to match. It is assumed that the matching characters begin in column 1. Subroutine URVDEF will not match the characters from the middle of the card. The characters to match must start from column 1.

APPENDIX B — System Dependent Routines (porting NESSUS to other systems)

The NESSUS code has been written using standard Fortran 77. However there are several subroutines that are system dependent. These are listed below and followed by a discussion of the system dependencies.

File	Description
dater	Subroutine to get the current date and time.
intint	Subroutine to open files. The record length for binary files may be system dependent.
nessus	NESSUS main program that has a flag to identify 32 or 64 bit architecture.
prompt	Subroutine to get the input data file name. Command line argument function may be system dependent.
quit	Subroutine to terminate execution. Prints final execution CPU time
timer	Subroutine to return elapsed CPU time.

dater

The system date and time functions for the specific system will need to be replaced in this routine.

intint

Depending on the computer system where NESSUS is executed, the file record length may need to be modified in the file open statements in intint.f. The NDBREC variable in intint.f defines the record length. The exact record length will depend on the computer system characteristics and the size of the data types employed in the program:

1. On typical 32-bit machines using 4-byte integers and 8-byte reals, the database record length should be set to 80-bytes.
2. On 32-bit VAX versions using 32-bit integers and 64-bit reals, the data base record length should be set to twenty 32-bit longwords.
3. On 64-bit architectures using 8-byte integers and 8-byte reals, the data base record length should be set to ten 64-bit words.

nessus

The IDP variable in nessus.f specifies how the integers and reals are stored. On typical 32-bit machines, this parameter is set to two since the double precision reals occupy two 32-bit integer words. On 64-bit machines, this value is set to one, since both integers and reals occupy a single 64-bit word.

prompt

The subroutine or function that returns the command line arguments is system dependent. The subroutine or function name is generally named GETARG or IGETARG and may have different number of arguments

quit

The elapsed CPU time function for the specific system will need to be replaced in this routine.

timer

The elapsed CPU time function for the specific system will need to be replaced in this routine.

APPENDIX C — NESSUS Interface Flowchart

This section describes the logic flow of the NESSUS code. It can be used as a starting point for adding interfaces to other finite element codes.

A flow chart of the NESSUS code is shown in Figure C-1.

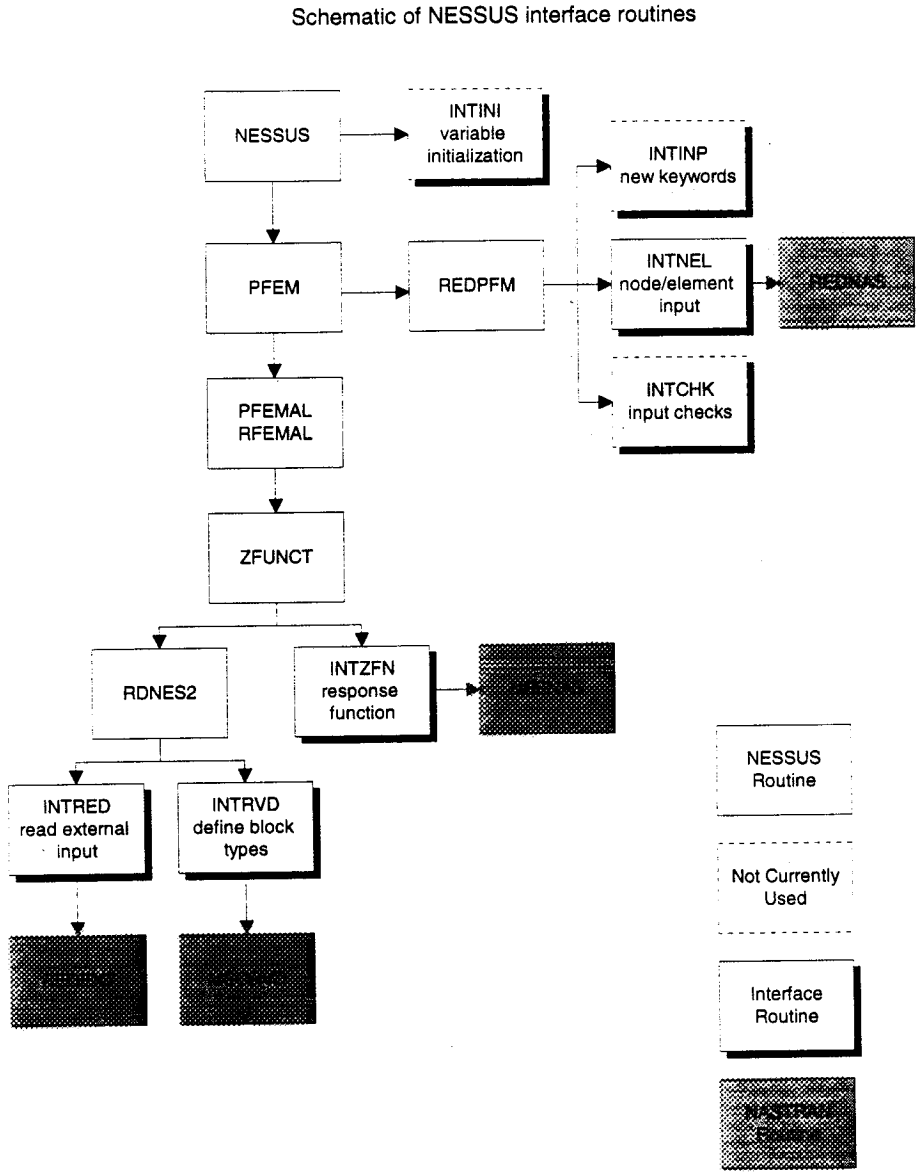


Figure C-1. NESSUS Flowchart

A description of the new routines developed for this interface is given in Table C-1.

Table C-1. Interface and NASTRAN Specific Subroutines	
File Heading	Description
cfield.f	User defined random variables - changes a specified field on a NASTRAN card
chngcard.f	Changes NASTRAN card for user defined random variables
iheadr.f	MSC routine for reading output2 file
intchk.f	Interface routine for interfaced code specific input data checks
intini.f	Interface routine for interfaced code specific variable initialization
intinp.f	Interface routine for interfaced code specific input
intnel.f	Interface routine for reading node and element numbers
intred.f	Interface routine to read external code input
intres.f	Interface routine for writing code specific results to the output file
intrvd.f	Interface routine for interfaced code specific DEFINE block types
intzfn.f	Interface routine for interfaced code specific execution response extraction
iopen.f	MSC routine for reading output2 file
iread.f	MSC routine for reading output2 file
lmatch.f	Matches a NASTRAN card
mscfield.f	Changes a NASTRAN card
mscpost.f	Routine for reading a MSC/NASTRAN output2 file
mscrvd.f	Identifies NASTRAN specific define block types (USER)
nasmatch.f	Finds matching NASTRAN bulk data card given *DEFINE card
nesmsc.f	Controls execution of NASTRAN and response extraction
parsa3.f	Parsing routine
parsa4.f	Parsing routine
pcard.f	Perturbs NASTRAN card
rdelnod.f	Interprets *NODE and *ELEM keywords (new and old format)
redext.f	Interface routine to read external analysis program input files
rednas.f	Identifies nodes and elements in the NASTRAN input (used to get a list of all nodes or elements for *NODE ALL or *ELEMENT ALL
urvdef.f	User subroutine for user define random variables
xcodei.f	Executes NASTRAN

APPENDIX D — Steps Necessary to Modify NESSUS for Other Finite Element Codes

Interface routines have been developed and documented to assist in integrating other finite element programs into NESSUS. There are four main parts to coupling an external finite element program with NESSUS for probabilistic analysis. These are

1. Read the external code specific input parameters used for defining random variables, responses etc.
2. Modify the code input to reflect the values of the random variables
3. Execute the external code
4. Extract the response of interest from the external code results file

The program structure is shown in Figure C-1 in Appendix C. The interface routines are identified and the MSC/NASTRAN specific routines provide an example of what is needed for interfacing to other external analysis programs. Table D-1. lists the interface routines with a brief description. It is anticipated that other finite element interfaces can be modeled directly off the NESNAS subroutine. This is the heart of the interface with MSC/NASTRAN and controls the input file creation, NASTRAN execution and response extraction.

Table D-1. Subroutines Specific to Interfacing NESSUS to the Finite Element Programs

File	Description
intchk.f	Interface routine for interfaced code specific input data checks
intini.f	Interface routine for interfaced code specific variable initialization
intinp.f	Interface routine for interfaced code specific input
intred.f	Interface routine to read external code input
intres.f	Interface routine for writing code specific results to the output file
intrvd.f	Interface routine for interfaced code specific DEFINE block types
intzfn.f	Interface routine for interfaced code specific execution response extraction

APPENDIX E — Updates to the PFEM Manual

Several pages of the PFEM manual need to be updated for the modified NESSUS code. The pages are given below.

*COMPUTATIONALMETHOD

```
*COMPUTATIONALMETHOD  icmod nsvars
jsvar(i),i=1,nsvars
```

The *COMPUTATIONALMETHOD keyword is used to select the computational method and the associated random variables. (Optional-use only if a finite element model is included.)

icmod is the computational method. (Required if nsvars is greater than zero)

icmod = 1 - NESSUS/FEM

icmod = 2 - NESSUS/BEM

icmod = 4 - MSC/NASTRAN

nsvars is the number of computational random variables. (Optional)

jsvar is an integer list of computational model random variable numbers. (Required if nsvars is greater than zero)

*NODE

```
*NODE
inode1 to inode2, inode3, inode4 to inode5,...

or

ALL
```

NODE selects the node numbers for the mean value analysis. This format is general. NESSUS will perform probabilistic analyses from inode1 to inode2, inode3, from inode4 to inode5, etc. Multiple lines of input are allowed. If the user selects "ALL", NESSUS will read all NASTRAN node numbers and perform a probabilistic analysis at all nodes.

*ELEM

```
*ELEM
ielem1 to ielem2, ielem3, ielem4 to ielem5,...

or

ALL
```

ELEM selects the element numbers for the mean value analysis. This format is general. NESSUS will perform probabilistic analyses from ielem1 to ielem2, ielem3, from ielem4 to ielem5, etc. Multiple lines of input are allowed. If the user selects "ALL", NESSUS will read all NASTRAN node numbers and perform a probabilistic analysis at all elements.

*NODE

```
*NODE
inode1 to inode2, inode3, inode4 to inode5,...
```

NODE selects the node numbers for the advanced mean value analysis. This format is general. NESSUS will perform probabilistic analyses from inode1 to inode2, inode3, from inode4 to inode5, etc. Multiple lines of input are allowed.

Specifying "ALL" is not allowed for AMV analysis.

*ELEM

```
*ELEM
ielem1 to ielem2, ielem3, ielem4 to ielem5,...
```

ELEM selects the element numbers for the advanced mean value analysis. This format is general. NESSUS will perform probabilistic analyses from ielem1 to ielem2, ielem3, from ielem4 to ielem5, etc. Multiple lines of input are allowed.

Specifying "ALL" is not allowed for AMV analysis.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	July 27, 1996	Final Report April 27, 1995 - July 27, 1996

4. TITLE AND SUBTITLE
NESSUS/NASTRAN Interface

5. FUNDING NUMBERS
NAS8-39797

6. AUTHOR(S)
Harry Millwater, David Riha

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Southwest Research Institute
6220 Culebra Road
PO Drawer 28510
San Antonio, TX 78228-0510

8. PERFORMING ORGANIZATION REPORT NUMBER
06-7212

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
National Aeronautics and Space Administration
George C. Marshall Space Flight Center
Marshall Space Flight Center, AL 35812

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

The NESSUS and NASTRAN computer codes were successfully integrated. The enhanced NESSUS code will use NASTRAN for the structural analysis and NESSUS for the probabilistic analysis. Any quantities in the NASTRAN bulk data input can be random variables. Any NASTRAN result that is written to the output2 file can be returned to NESSUS as the finite element result. The interfacing between NESSUS and NASTRAN is handled automatically by NESSUS. NESSUS and NASTRAN can be run on different machines using the remote host option.

14. SUBJECT TERMS
probabilistic finite element, NASTRAN, NESSUS, structural reliability

15. NUMBER OF PAGES
29

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	SAR